ARGUMENTS ACCOMPANYING APPLICANTS' REQUEST FOR PRE-APPEAL REVIEW

I. Status of Claims

Claims 1-33 are currently pending with claims 1, 6, 9, 14, 19, 24, and 29 being independent. No claims have been allowed by the Examiner.

II. Summary of various claimed features of the present application

As discussed at length in the specification and in the five responses previously filed by Applicants, embodiments of the present invention enable queries in a query language, such as a SQL query, to be initially associated with a declarative language function, such as a LISP function, which in turn may be associated with an imperative language statement, such as a C++ statement (see paragraph 0023). Such functionality enables portable, handheld, and other limited processing devices that are unable to employ query languages to nevertheless execute queries utilizing imperative language statements.

III. The errors in the Examiner's obviousness rejections of all independent claims

The Examiner contends that the combination of Xu (U.S. Patent No. 5,995,958) and Laitinen (U.S. Patent No. 5,862,383) discloses all features recited in the claims of the present application. As is discussed at length below, Xu discloses a stand-alone query language (i.e. a replacement for SQL) and Laitinen discloses a digital signal processing (DSP) language that includes declarative and imperative components. No combination of Xu and Laitinen provide any disclosure or suggestion relating to the conversion of queries from a query language to any other language. Thus, the combination of Xu and Laitinen does not disclose or suggest features recited in every independent claim of the present application, including associating declarative language representations with queries in a query language.

A. Xu and Laitinen do not disclose or suggest receiving queries or interpreting queries

The Examiner's § 103 rejections are founded on the premise that Xu discloses both (1) receiving queries in a query language and (2) interpreting received queries by associating at least one declarative language function with the terms that comprise the queries (Office action, pages

2-3). Xu does not disclose or suggest either of these features as Xu merely provides a new database query language based on lambda calculus and does not disclose receiving queries in a query language and converting queries to other languages through an association with declarative language functions.

Xu's lambda-calculus query language allows database management systems to have full computing capabilities without the need for application-dependent software development in a host programming language (Abstract). Thus, by using only the lambda-calculus query language, the database disclosed by Xu can perform high-level functions unable to be performed by conventional query languages like SQL (col. 6, ll. 5-23). Xu does not disclose receiving conventional queries (such as SQL queries) and converting them to the lambda-calculus language, as Xu requires the queries to be initially written in the lambda-calculus language (col. 13, ll. 11-25). Xu also does not disclose or suggest converting its lambda-calculus queries to other languages, such as declarative or imperative languages, as the lambda-calculus query language is intended to be the only language needed (col. 4, ll. 46-65; col. 6, ll. 5-23).

The Examiner cites Xu, col. 2, as disclosing the claimed feature of receiving a query in a query language. However, Xu, cols. 1-2, is merely discussing the operation of a conventional relational database system (col. 1, ll. 57-58) and the disadvantages associated therewith (col. 2, ll. 50-51). Xu's query language enables high-level functionality not provided by conventional query languages such as SQL (col. 4, ll. 46-65). Thus, Xu has no use for conventional queries or the association of declarative language functions therewith.

The Examiner cites Xu, cols. 4 and 13-14, as disclosing the association of declarative languages with queries in query language. However, these portions of Xu only disclose allowing a programmer to write and form queries using a calculus-based query language and not the association of declarative language functions (such as a LISP function) with a query (such as a SQL query) (col. 4, ll. 41-45; col. 13, ll. 11-25; col. 13, ll. 42-45; col. 14, ll. 12-13; col. 14, ll. 22-23; etc). As Xu's lambda-calculus query language is operable to perform various high-level functions, Xu provides no reason to convert queries to declarative and imperative language functions (col. 4, ll. 46-65). Thus, Xu expressly teaches away from converting queries in a query language to declarative or imperative languages.

In contrast, embodiments of the present invention provide a method for allowing computing devices (such as PDAs) to manage or interact with a database even though they are unable to execute query language statements (which would likely include Xu's calculus-based query language). Conceivably, embodiments of the present invention could be employed to convert Xu's query language to imperative language statements for execution by a low-power computing device.

Laitinen, alone or in combination with Xu, also does not disclose or suggest receiving queries or converting queries to other languages as Laitinen's DSP teachings have absolutely no relevance to databases or database queries.

B. Xu and Laitinen do not disclose or suggest converting queries represented by declarative language statements

The Examiner concedes that Xu does not disclose or suggest various features claimed by the present application such as the "converting the queries represented by the at least one declarative language function to a plurality of imperative language statements" feature recited in claim 1. The Examiner relies on the teachings of Laitinen in combination with Xu to attempt to render all pending claims obvious.

Laitinen does not disclose or suggest any features recited in the claims of the present application. Laitinen is directed at a method for generating code for a digital signal processor and its teachings have no relevance to databases or database queries (Abstract; col. 1, ll. 5-10).

The portions of Laitinen cited by the Examiner discuss the conversion of technical specifications of a programming language between imperative and declarative forms of the programming language (col. 3, ll. 18-26). The "technical specifications" which are converted are not queries and do not relate to query languages and are instead "communication protocol specifications which control the encoding and decoding functions" of the digital signal processor (col. 3, ll. 26-30). Thus, Laitinen discloses converting communication protocol specifications between two levels of one programming language (TSN) to enable varying control of the digital signal processor. Laitinen does not disclose or suggest converting queries represented by declarative language functions to imperative language statements.

As such, no combination of the prior art references discloses or suggests converting queries in a query language to any other language—declarative or imperative. Combining Xu's query language with Laitinen's DSP method does not result in a system that allows queries in

Xu's query language to be associated with declarative language functions and then converted to Laitinen's imperative language as neither Xu nor Laitinen provide any query conversion.

The dependent claims recite further features neither taught nor suggested by the Examiner's combination. For example, claims 2, 15, and 25 recite "converting the query language to an intermediate tree representation corresponding to the at least one declarative language function associated with the plurality of query terms, and thereafter converting the query to at least one data structure that is interpreted by an imperative language interpreter core to perform the queries," which is neither taught nor suggested by any reference of record. The Examiner fails to identify any portions of Xu or Laitinen that disclose or suggest this feature.

III. The errors in the Examiner's 35 U.S.C. § 101 rejections

The Examiner rejected claims 1, 6, 9, 14, 24, and 29, for being directed at non-statutory subject matter as they allegedly do not produce a useful, tangible, and concrete result. In particular, the Examiner concludes, without any supporting analysis or citations, that "software per se" is unpatentable subject matter unless the software stores computed results in a physical medium (Office action, page 2).

The Examiner's brief § 101 analysis violates both the Interim Guidelines and controlling law. In particular, the Examiner's requirement that claimed subject matter produce a physical transformation through the storage of information within a memory is clearly wrong and contrary to the law. Assuming *arguendo* that the claimed subject matter does not induce a physical transformation, the requirements of § 101 may still be met if the final result achieved is "useful, tangible, and concrete" (see Interim Guidelines, page 20).

The result achieved by the claimed subject matter is certainly useful—it allows computing devices to easily execute queries in languages other than query languages. The achieved result is also tangible as it produces a real world result—interaction with a database by executing imperative language statements generated from queries in a query language. Contrary to what is demanded by the Examiner, the "tangible requirement" does not require claimed subject matter to be tied to a particular apparatus such as a memory (see Interim Guidelines, page 21). The achieved result is also concrete as it is easily repeatable and is not dependent upon a particular query input or condition (see Interim Guidelines, page 21). Thus, the claims of the present application clearly are directed at statutory subject matter as defined by the Interim Guidelines.

The half-dozen Office actions issued by the Examiner lack the analysis required by the Interim Guidelines and controlling law. It is the Examiner's burden to establish a *prima facie* case of unpatentability under § 101 and the Examiner has failed to satisfy that burden.

IV. Conclusion

In view of the forgoing, Applicants respectfully submit that the Examiner has failed to form a prima facie case of obviousness or unpatentability and all claims are allowable as a result. Should any questions remain, please contact the undersigned. Any additional fee which might be due in connection with this application should be applied against our Deposit Account No. 19-0522.

Respectfully submitted,

HOVEY WILLIAMS LLP

BY: [illegible]

Scott R. Brown, Reg No. 40,535
2405 Grand Blvd., Suite 400
Kansas City, Missouri 64108
(816) 474-9050

ATTORNEYS FOR APPLICANTS